ISSN: 2472-3703 DOI: 10.6911/WSRJ.202501_11(1).0002

Synthesizer Based Efficient Self-Attention for Vision Tasks

Guangyang Zhu[1, a], Jianfeng Zhang[2, b, *], Yuanzhi Feng[3, c] and Hai Lan[4, d]

[1]School of Geospatial Information, Information Engineering University, Zhengzhou, China

[2]Department of Information, East China Normal University, Shanghai, China

[3]Software Engineering Institute, East China Normal University, Shanghai, China

[4]Fujian Institute of Research on the Structure of Matter, Chinese Academy of Sciences, Quanzhou, China

[a]zhuguangyang13@126.com, [b]zhangjf1995@gmail.com, [c]fengyuanzhi21@mails.ucas.ac.cn, [d]lanhai09@fjirsm.ac.cn

[*] Corresponding author

Abstract

Attention mechanism was first designed for natural language processing (NLP) and then was widely applied in the field of computer vision, which shows notable competence in capturing long-range relationships. However, the dot product multiplication among query-key-value features within the self-attention module results in exhaustive and redundant computation. It is impractical for a self-attention module to directly handle raw image data with millions of pixels. As a result, an image is usually partitioned into a sequence of small patches or is processed by a Convolutional Neural Network backbone to make the computation tractable before feeding into a self-attention module. Furthermore, dimension alignment among query-key-value features within the self-attention module might destroy the internal structure of the visual feature maps. To address these problems, this paper proposes a plug-in module named Synthesizing Tensor Transformations (STT) with its variants for self-attention which directly processes pixel-level image features. Instead of computing the dot-product multiplication among query-key-value, the basic STT learns to obtain the synthetic attention weight by transforming the input visual tensor. The effectiveness of STT series is validated on the image classification and image captioning. Experiments show that the proposed STT achieves competitive performance while keeping robustness compared to basic self-attention.

Keywords

Synthesizer; Self-Attention; Efficient Visual Transformer; Image Classification; Image Captioning.

1. INTRODUCTION

In recent years, the attention mechanism has generated considerable interest in the domain of deep learning. The main breakthroughs of attention modules first appeared in Natural Language Processing (NLP) [1–6], and then also in computer vision domain [7–10]. These achievements have demonstrated that the attention module provided a different approach than Convolutional Neural Network (CNN) models to handle the task and demonstrated a promising performance.

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202501_11(1).0002

Generally, the attention module tends to exert a learnable weight on features to distinguish importance from various perspectives. According to the computing techniques and tasks, the attention modules can be categorized into two types. Hard attention module is non-differentiable and requires computation tricks [11,12], while soft attention module is differentiable thus having wider applications for visual tasks [13–15]. As a particular form of attention, self-attention is applied as the core mechanism of the neural network named transformer, which is widely employed in the field of computer vision. The essential advantage of self-attention is that it focuses on the relative importance of own content rather than weighing across multiple contents as a general attention mechanism. As one of the critical roles in the self-attention mechanism, dot product multiplication among the features of query, key, and value plays a significant role in learning self-alignment [16]. Precisely, employing the dot product between a single token and all other tokens in the sequence could formulate the relative importance score. Through pairwise dot product, the self-attention mechanism establishes a content-based retrieval process.

However, do we really need to use dot production in the self-attention mechanism? Is it the best choice to learn the self-alignment between the features of query and key? We cannot deny that attention-based architecture is one of the most efficient models in machine learning, but it also has space to improve [17]. Investigating the alternative plug-in module for current architecture could help us to have a deeper understanding of the attention mechanism. Undoubtedly, dot product multiplication has several benefits. But it also induces exhaustive and redundant computation while exerting on the features. Especially on large-scale processing data, the extra parameters and memory caused by dot product operations enormously increase the burden of the training process. Moreover, in the computer vision task, self-attention for images tends to compute the similarity scores among visual features [7,15,18] rather than mapping for retrieval. In order to overcome the shortages of dot production, it is necessary to search for another approach to improve the performance of the self-attention mechanism. As a result, recent findings regarding the NLP attention module have led to an alternative approach, named Synthesizer Attention [16], as shown in Figure 1.

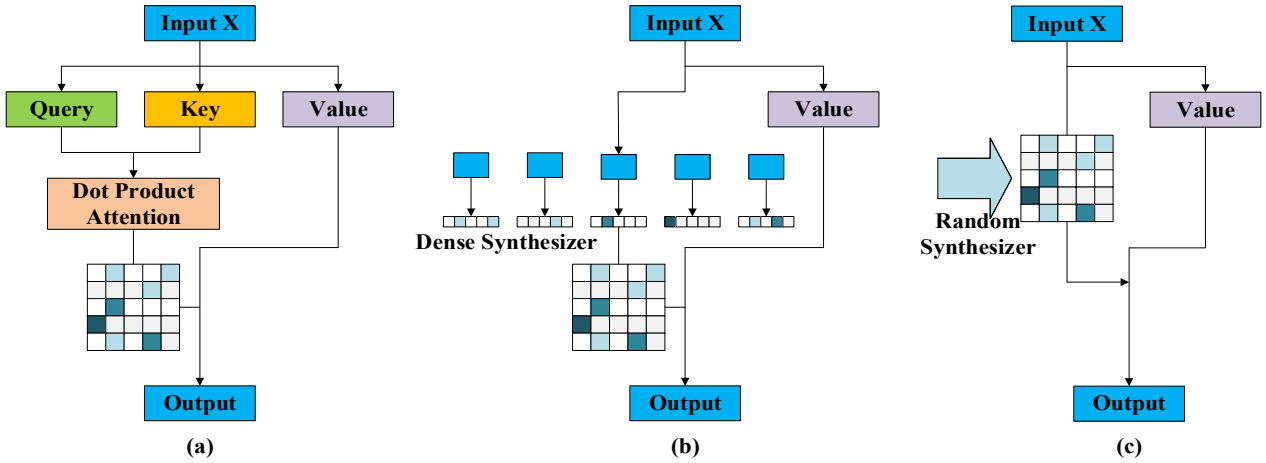

Figure 1. Dot product attention VS. synthesizers for NLP [16]. (a) Dot product based self-attention for NLP. (b) Dense Synthesizer for NLP. (c) Random Synthesizer for NLP.

In this work, we provide a viable alternative to overcome the shortage brought from the dot-production in the standard self-attention module, to pursue a better performance of visual self-attention. Inspired by the NLP-like synthesizer [16,19], we propose a plug-in synthesizer to replace traditional visual self-attention modules, namely, the Visual Dense Synthesizing (VDS)

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202501_11(1).0002

module. Compared with the NLP-like synthesizer, which contains one-dimensional natural language vectors, our module applies the multidimensional tensor to adapt the presentation of visual features. The module is basically composed of a tensor transformation for self-alignment between query matrix and key matrix, without computing the dot-product multiplication among the features of query-key-value. Removing dot product operations, the VDS gains some straightforward advantages:

1. Replacing dot production with a learnable linear synthesizer reduces dependency on the input, thus enhancing the robustness of the model against external perturbations;

2. The proposed linear synthesizer can be easily adapted to input data structure, without reshaping the raw input. Thus, it is easier to preserve the underlying structure of feature maps;

3. Since the alignment transformations in the synthesizer are computed upon various dimensions of feature maps, the attention is produced simultaneously in both channel and spatial domain.

To further simplify the linear transformations in synthesizing, derived from the regular VDS module, the paper further proposes a series of plug-and-play modules, namely, Synthesizing Tensor Transformations (STT). The STT series are finally applied for the image classification task and image caption task. According to the result of our typical classification experiment, the STT series demonstrates that it is not only a viable alternative for the traditional self-attention module but also has better robustness.

2. RELATED WORK

2.1. Self-Attention Mechanism

Self-attention attracts considerable interest due to its versatile application. For CNN-based models, self-attention mechanism has been used in many modules, such as extra re-weighting modules for channels [13,14,20], jointly spatial and channel attention module [15,21,22], or remold convolution operation with self-attention [23–26].

Another line of works arranges self-attention to be a component in a pipeline for downstream tasks, such as augmentation of feature maps for classification [23,27], object detection [7,28,29], segmentation [30], image captioning [31] and depth estimation [32]. However, it is difficult to directly apply the self-attention mechanism to pixel-wise data. [33] restricted self-attention convolution within local neighborhoods for query pixels, and [30,34] restricted computation along individual axes. [35] reduced image resolution and color space before transformer, and [36] directly applied transformer on image sequence patches.

To reduce computational costs, some of the work adjust the structure of attention. [37] divided and permuted the feature map thus smaller attention map would be conquered and permuted back. [38] focused on the criss-cross around the key points on the feature of query, key and value. [39] adopted double attention to re-allocate features for channels and the total workload shrinks after feature gathering. Differently, another way is to reduce the length at certain dimension [40], or divide computation by locality sensitive hashing [3].

2.2. Dot Product in Self-Attention

The regular dot-product operator based self-attention serves as a basic building block of vision tasks [23–26].

As is illustrated in Figure 2, let $\mathcal{X} \in \mathbb{R}^{H \times W \times C}$ be an input tensor with the height H, the width W, the number of channels C. Before being fed into the self-attention block, the 3D tensor $\mathcal{X}$ is first flattened to a 2D matrix $X \in \mathbb{R}^{HW \times C}$.Then input X is projected to corresponding representations Queries $Q \in \mathbb{R}^{HW \times C}$, Keys $K \in \mathbb{R}^{HW \times C}$ and Values $V \in \mathbb{R}^{HW \times C}$ with trainable transformations such as linear mappings or convolutions, and have matching output

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202501_11(1).0002

dimensions. Take linear mapping as an example, the projection can be represented with 2D matrices $W_Q \in \mathbb{R}^{C\times d}$, $W_K \in \mathbb{R}^{C\times d}$ and $W_V \in \mathbb{R}^{C\times d}$, respectively. And d is the desired output dimension of Q, K and V. The corresponding representations of Queries Q, Keys K and Values V can be computed by:

$$\begin{aligned} \mathrm{Q}&=\mathrm{XW_Q}, \\ \mathrm{K}&=\mathrm{XW_K}, \\ \mathrm{V}&=\mathrm{XW_V} \end{aligned} \tag{1}$$

The attention coefficients $S \in \mathbb{R}^{HW\times HW}$ can be obtained by dot product operation:

$$S=\mathrm{Softmax}\left(\frac{QK^{\top}}{\sqrt{d}}\right), \tag{2}$$

where S_{ij} measures the similarity between the i^{th} row of Q and the j^{th} row of K. Finally, the output $Y \in \mathbb{R}^{HW\times C}$ is the weighted average over Values V with coefficients S:

$$Y=\mathrm{Attention}(Q,K,V)=SV. \tag{3}$$

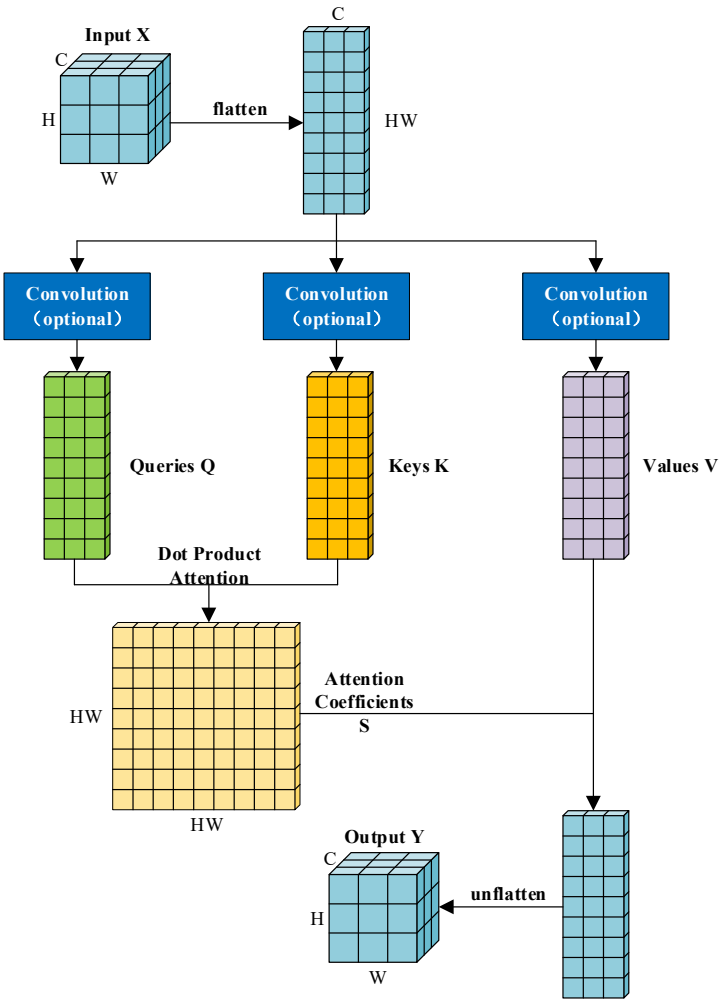

Figure 2. Dot-product visual self-attention.

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202501_11(1).0002

3. SYNTHESIZING TENSOR TRANSFORMATION

To introduce our Synthesizing Tensor Transformation, we first give a fundamental model of tensor synthesizer. And then, the specific tensor synthesizers would be provided as (a) Tensor Dense Synthesizer, (b) Tensor Random Synthesizer, and (c) Tensor Factorized Synthesizer.

3.1. Preliminaries

Basic Tensor Synthesizer In the case of regular visual transformation architecture [36], the tensor input $\mathcal{X} \in \mathbb{R}^{H \times W \times C}$ is reshaped into 2D matrix $X \in \mathbb{R}^{HW \times C}$ and the dot-product self-attention could be interpreted as producing a dimension alignment from $\mathbb{R}^{HW \times C}$ to $\mathbb{R}^{HW \times HW}$, as is shown in Figure 2.

However, the flattening of input tensor may results in the loss of spatial information and a prohibitively expensive computation when the dimension is high. As a result, to directly process the tensor input $\mathcal{X} \in \mathbb{R}^{H \times W \times C}$, we propose to synthesize tensor transformation for replacing the dot-product self-attention in this section.

Formally, let $\mathcal{Z} \in \mathbb{R}^{H \times W \times d}$ be the projected features of $\mathcal{X} \in \mathbb{R}^{H \times W \times C}$ via the mapping function $\mathcal{F}(\cdot)$, where C and d are the number of channels.

$$\mathcal{Z} = \mathcal{F}(\mathcal{X}) \tag{4}$$

Intuitively, $\mathcal{F}(\cdot)$ could be a 3-mode tensor product. In detail, the mapping function $\mathcal{F}(\cdot)$ could be implemented as a three-layered feed-forward network, as follows:

$$\mathcal{F}(\mathcal{X}) = \mathcal{X} \times_H \mathbf{A}_{(H)} \times_W \mathbf{A}_{(W)} \times_C \mathbf{A}_{(C)}, \tag{5}$$

with $\mathbf{A}_{(H)}$, $\mathbf{A}_{(W)}$, and $\mathbf{A}_{(C)}$ are three matrices. For the convenience of understanding the tensor product, Figure 3 gives an example of n-mode production. For a detailed introduction of tensor products, we refer the interested readers to [41-43].

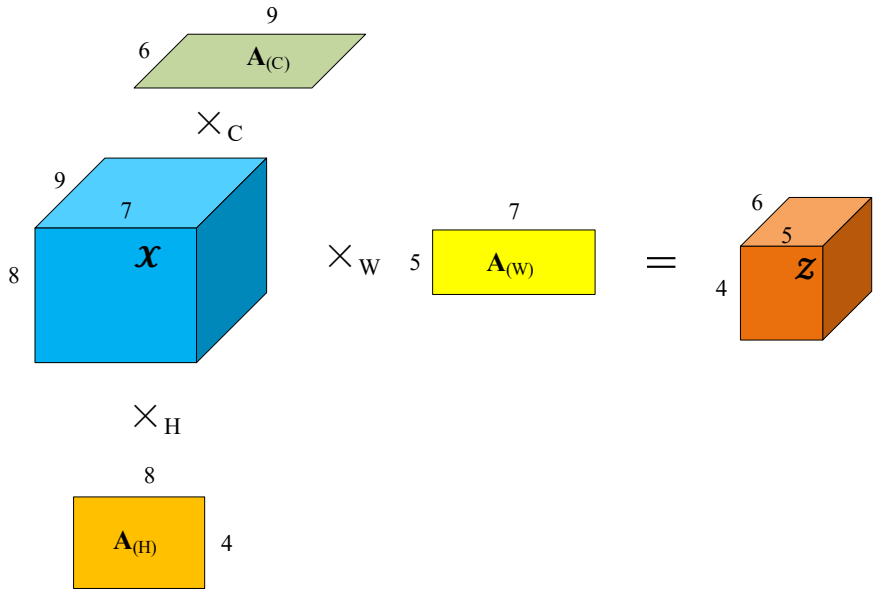

Figure 3. This figure visualizes an example of the $n-$mode product. The third-order tensor $\mathcal{X} \in \mathbb{R}^{8 \times 7 \times 9}$ is mapped into $\mathcal{Z} \in \mathbb{R}^{4 \times 5 \times 6}$ with a 3-mode product using matrices $\mathbf{A}_{(H)} \in \mathbb{R}^{4 \times 8}$, $\mathbf{A}_{(W)} \in \mathbb{R}^{5 \times 7}$, $\mathbf{A}_{(C)} \in \mathbb{R}^{6 \times 9}$.

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202501_11(1).0002

Definition 1 (n−mode product). *Given a tensor* $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ *and a matrix* $\mathbf{A}_n \in \mathbb{R}^{J_n \times I_n}$ *, the n-mode product is denoted by:*

$$\mathcal{X} \times_n \mathbf{A}_n, \tag{6}$$

and results in an $I_1 \times I_2 \times \cdots \times I_{n-1} \times J_n \times I_{n+1} \times \cdots \times I_N$ *tensor. The entries of this tensor are defined as:*

$$\left(\mathcal{X} \times_n \mathbf{A}_n\right)_{i_1 i_2 \cdots i_{n-1} j_n i_{n+1} \cdots i_N} = \sum_{i_n=1}^{I_n} x_{i_1 i_2 \cdots i_{n-1} i_n i_{n+1} \cdots i_N} \cdot a_{j_n i_n}, \tag{7}$$

for all $j_n = 1, \cdots, J_n$.

Besides, to obtain an appropriate size of the matrix, a method named *n-mode matrix unfolding* is required. Depending on this method, the multiplication is conducted between a k-order tensor $\mathcal{X}$ along all its modes and matrices $A_1, \cdots, A_k$ [43] and the layered feed-forward network could be built as follows:

$$\mathcal{Y} = \mathcal{X} \times_1 A_1 \times_2 A_2 \times_3 \cdots \times_k A_k. \tag{8}$$

It can be rewritten as the following linear system:

$$\text{vec}(\mathcal{Y}) = \left(A_k \otimes \cdots \otimes A_2 \otimes A_1\right) \text{vec}(\mathcal{X}), \tag{9}$$

where $vec(\cdot)$ means the vector space isomorphism: $\mathbb{R}^{I_1 \times I_2 \times \cdots \times I_k} \to \mathbb{R}^{I_1 I_2 \cdots I_k}$, which denotes the operator that unfolds a tensor along the last dimension [43]. Therein, $\otimes$ denotes the Kronecker product operator. with which the Kronecker product of A and B is defined by:

$$\mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} A_{11}\mathbf{B} & \cdots & A_{1k_1}\mathbf{B} \\ \vdots & \ddots & \vdots \\ A_{a1}\mathbf{B} & \cdots & A_{ak_1}\mathbf{B} \end{bmatrix}, \tag{10}$$

with A_{ij} being an entry in **A**. More details about the Kronecker product could be found in [42]. Therefore, the mapping $\mathcal{F}(\cdot)$ could be presented as the tensor multiplication as the Eq. (5). Figure 4 shows the process of the tensor multiplication with Kronecker product operator.

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202501_11(1).0002

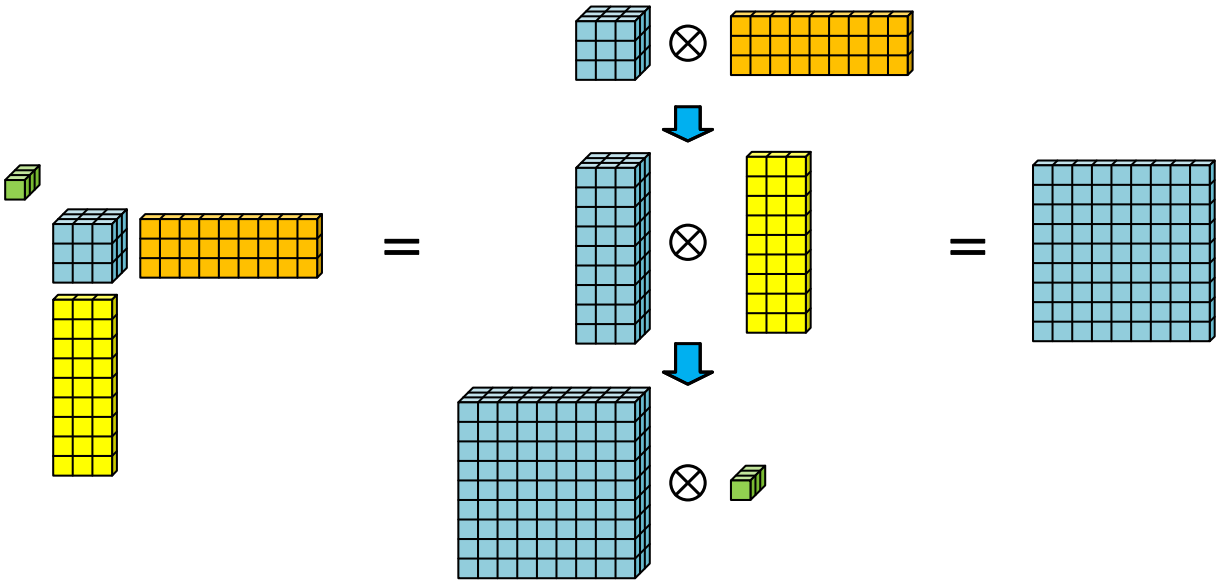

Figure 4. Illustration for tensor multiplication with Kronecker product operator. In this case, a three-dimensional tensor is first expanded along its vertical dimension. Then, the horizontal dimension is expanded. After that, the third dimension is reduced to get a two-dimensional matrix.

3.2. Synthesizer Tensor Model

Tensor Dense Synthesizer To achieve the mentioned dimension alignment on raw tensor $\mathcal{X}$ without the dot-product self-attention mechanism, we define a tensor transformation to compute attention coefficients by replacing the dot-product self-attention.

Definition 2 (Tensor transformation). *Given the input tensor* $\mathcal{X} \in \mathbb{R}^{H\times W\times C}$, *the attention coefficients defined in Eq. (2) could be computed with n-mode production:*

$$Z = \mathcal{X} \times_1 \mathbf{A}_{(H)}^{(l)} \times_2 \mathbf{A}_{(W)}^{(l)} \times_3 \mathbf{A}_{(C)}^{(l)}, \tag{11}$$

$$S = \text{Softmax}\left(Z\right), \tag{12}$$

with $\mathbf{A}_{(H)}^{(l)} \in \mathbb{R}^{HW\times H}$, $\mathbf{A}_{(W)}^{(l)} \in \mathbb{R}^{HW\times W}$, $\mathbf{A}_{(C)}^{(l)} \in \mathbb{R}^{1\times C}$, $Z, S \in \mathbb{R}^{HW\times HW}$, *and* l *denotes the layer number.*

The combination of Eq. (11) and Eq. (12) is referred to the VDS module, which eliminates the dot product altogether by replacing $QK^{\top}$ in standard self-attention with the synthesizing function defined in Eq. (11), as shown in Figure 5(**a**). Then, the output is computed by:

$$Y = \text{Synthesizer}(Q, V) = SV \tag{13}$$

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202501_11(1).0002

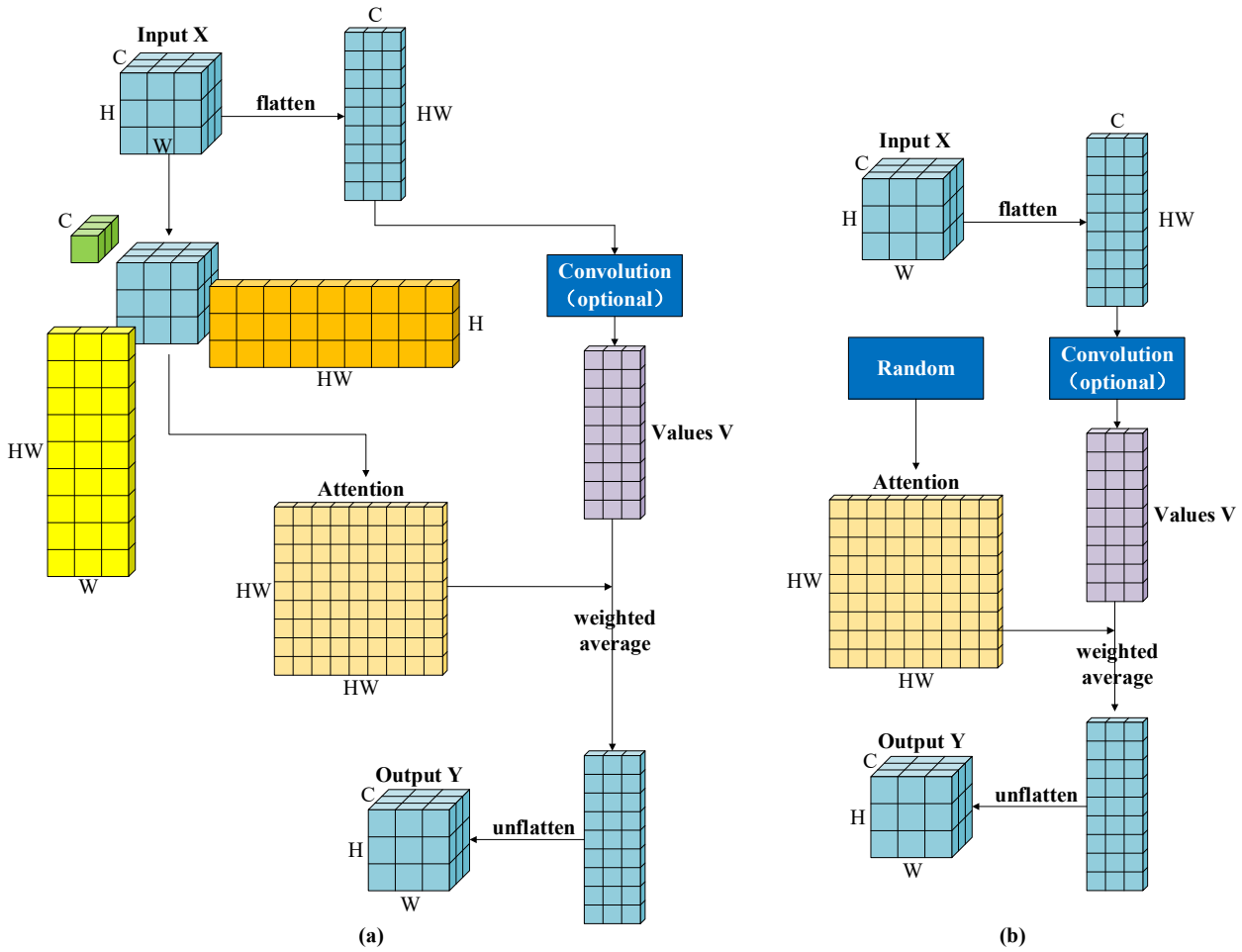

Figure 5. Phase analysis of arrangement

However, to achieve dimension alignment on input tensor $\mathcal{X}$ with Eq. (11), the output tensor $Z \in \mathbb{R}^{HW \times HW}$ could be assigned the dimension with three different transformations, as shown in Figure 6. Therein, Z_H, Z_W, Z_C are computed as follows:

$$Z_H = \mathcal{X} \times_1 \mathbf{A}_{(1)} \times_2 \mathbf{A}_{(2)} \times_3 \mathbf{A}_{(3)}, \tag{14}$$

with $\mathbf{A}_{(1)} \in \mathbb{R}^{1 \times H}$, $\mathbf{A}_{(2)} \in \mathbb{R}^{HW \times W}$, $\mathbf{A}_{(3)} \in \mathbb{R}^{HW \times C}$;

$$Z_W = \mathcal{X} \times_1 \mathbf{A}_{(1)} \times_2 \mathbf{A}_{(2)} \times_3 \mathbf{A}_{(3)}, \tag{15}$$

with $\mathbf{A}_{(1)} \in \mathbb{R}^{HW \times H}$, $\mathbf{A}_{(2)} \in \mathbb{R}^{1 \times W}$, $\mathbf{A}_{(3)} \in \mathbb{R}^{HW \times C}$;

$$Z_C = \mathcal{X} \times_1 \mathbf{A}_{(1)} \times_2 \mathbf{A}_{(2)} \times_3 \mathbf{A}_{(3)}, \tag{16}$$

with $\mathbf{A}_{(1)} \in \mathbb{R}^{HW \times H}$, $\mathbf{A}_{(2)} \in \mathbb{R}^{HW \times W}$, $\mathbf{A}_{(3)} \in \mathbb{R}^{1 \times C}$.

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202501_11(1).0002

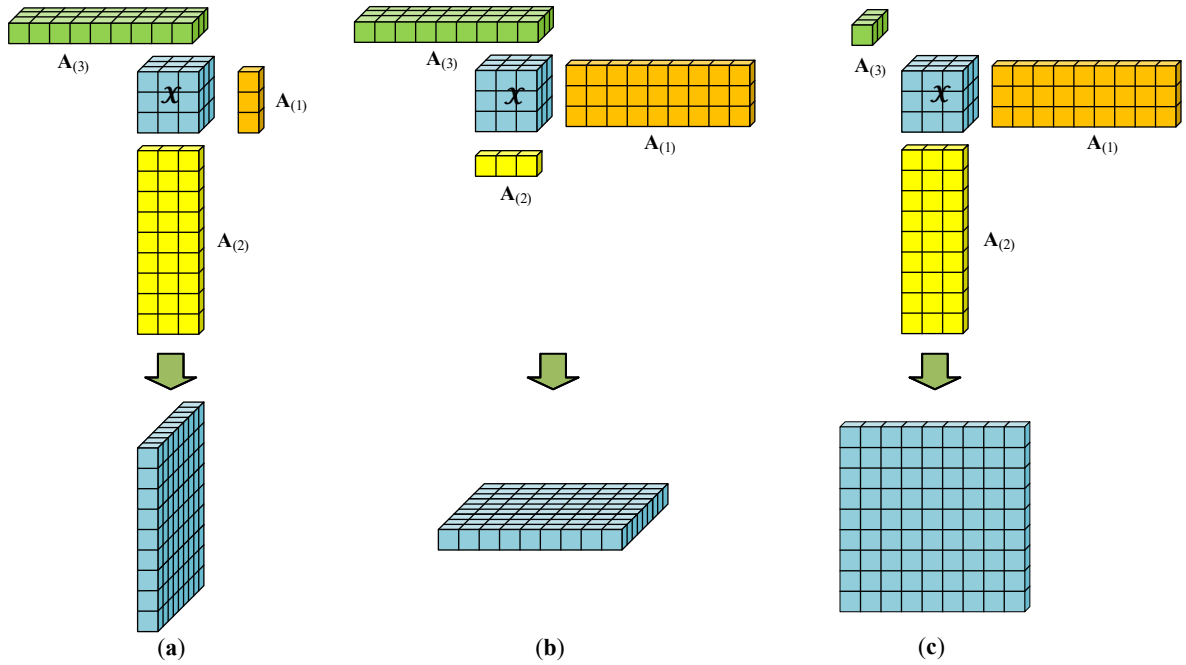

Figure 6. Three different transformations that achieve the same dimension alignment: (**a**) reducing the first dimension, (**b**) reducing the second dimension or (**c**) reducing the third dimension, while expanding the other two dimensions.

It is difficult to determine the dimensions of $\mathbf{A}_{(H)}^{(l)}$, $\mathbf{A}_{(W)}^{(l)}$, $\mathbf{A}_{(C)}^{(l)}$, i.e., how to assign the dimensions HW, H, W, 1 to three transformations. We note that all the proposed synthetic attention variants can be mixed in an additive fashion. As a result, Eq. (13) can be expressed as:

$$Y = \text{Softmax}\left(Z_H + Z_W + Z_C\right)V. \tag{17}$$

Tensor Random Synthesizer In the model of Tensor Dense Synthesizer, the features of input $\mathcal{X}$ are projected from $\mathbb{R}^{H\times W\times C}$ to $\mathbb{R}^{HW\times HW}$ via the 3-mode tensor product. Intuitively, the synthesizer implements the conditioning on each dimension independently. In contrast, the Tensor Random Synthesizer utilizes the random value as initialized attention weights, as shown in Figure 5(**b**). Then, the attention weights could either follow the training process to update, or keep fixed.

Since the synthesizer can be regarded as a self-alignment of dimensions, we can try to set Z as a random matrix R and the Random Synthesizer is defined as:

$$Y = \text{Synthesizer}(Q,V) = \text{Softmax}\left(R\right)V, \tag{18}$$

with $R \in \mathbb{R}^{HW\times HW}$ being a randomly initialized matrix. The attempt to utilize the random initial matrix identifies the idea that it is unnecessary to rely on any information from the individual input feature. Therefore, the Tensor Random Synthesizer tends to learn the task-specific alignment through the training process.

Compared with the token-token self-attention, the synthesizers can handle the input tensors with longer sequences or higher dimensionality. In our work, the factorized approach will further reduce computation, and we will introduce it in the next section.

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202501_11(1).0002

3.3. Multiple-level Factorized Tensor Synthesizer

We replace the dot-production in the standard self-attention module with our synthesizer module in the Tensor Dense Synthesizer. However, the synthesizer is too burdensome to learn when the size of parameters is too large. As shown in Eq. (11), the basic 3-mode tensor synthesizer contains several parameters $\mathbf{A}_{(H)}^{(l)} \in \mathbb{R}^{HW\times H}$, $\mathbf{A}_{(W)}^{(l)} \in \mathbb{R}^{HW\times W}$, $\mathbf{A}_{(C)}^{(l)} \in \mathbb{R}^{1\times C}$. Therefore, there are several factorized tensor synthesizer models introduced.

Tensor Dense Factorized Synthesizer To reduce the size of parameters, partially avoiding the over fitting while HW being huge, the parameter matrices within the Kronecker product could be decomposed [42], as shown in Figure 7. The Kronecker decomposition is the inverse process of Kronecker product operation.

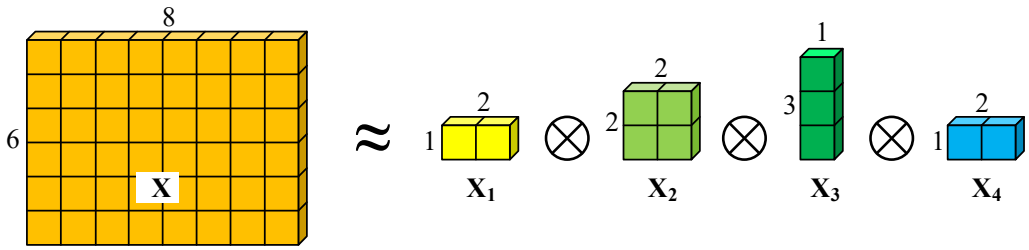

Figure 7. An example of Kronecker Decomposition. As the inverse process of Kronecker production, $X \in \mathbb{R}^{6\times 8}$ could be decomposed into $X_1 \in \mathbb{R}^{1\times 2}$, $X_2 \in \mathbb{R}^{2\times 2}$, $X_3 \in \mathbb{R}^{3\times 1}$ and $X_4 \in \mathbb{R}^{1\times 2}$, where $6 = 1\times 2\times 3\times 1$ and $8 = 2\times 2\times 1\times 2$.

For example, $\mathbf{A}_{(H)}^{(l)}$ could be approximated by N small matrices:

$$\mathbf{A}_{(H)}^{(l)} \approx \mathcal{D}\left(\mathbf{A}_{(H)}^{(l)}\right) = \mathbf{A}_{(H)_1}^{(l)} \otimes \mathbf{A}_{(H)_2}^{(l)} \otimes \cdots \otimes \mathbf{A}_{(H)_N}^{(l)}, \tag{19}$$

where $\mathcal{D}(\cdot)$ represents Kronecker decomposition, $\mathbf{A}_{(H)_i}^{(l)} \in \mathbb{R}^{\alpha_i\times\beta_i}$ for $i = 1, 2, \cdots, N$, $HW = \alpha_1\alpha_2\cdots\alpha_N$, $H = \beta_1\beta_2\cdots\beta_N$. By using the Kronecker decomposition, $\mathbf{A}_{(H)}^{(l)}$, $\mathbf{A}_{(W)}^{(l)}$, $\mathbf{A}_{(C)}^{(l)}$ can be easily factorized to several very small matrices. The Tensor Dense Synthesizer in Eq. (13) could be reformulated as Tensor Dense Factorized Synthesizer, which is defined as:

$$\begin{aligned} \mathcal{Z} &= \mathcal{X} \times_H \mathcal{D}_H\left(\mathbf{A}_{(H)}^{(l)}\right) \times_W \mathcal{D}_W\left(\mathbf{A}_{(W)}^{(l)}\right) \times_C \mathcal{D}_C\left(\mathbf{A}_{(C)}^{(l)}\right), \\ \mathcal{S} &= \text{Softmax}(\mathcal{Z}), \\ Y &= \text{Synthesizer}(Q, V) = \mathcal{S}V, \end{aligned} \tag{20}$$

where $\mathcal{D}(\cdot)$ decomposes the weight matrices.

Tensor Random Factorized Synthesizer Similarly, the factorized synthesizer could also be applied into Tensor Random Synthesizer, as shown in Figure 7. The idea is to decompose the random coefficient matrix $R \in \mathbb{R}^{HW\times HW}$ into several low rank matrices $R_i \in \mathbb{R}^{\alpha_i\times\beta_i}, i = 1, 2, \cdots, N$:

$$\begin{aligned} \mathcal{R} &= R_1 \otimes R_2 \otimes \cdots \otimes R_N, \\ Y &= \text{Synthesizer}(Q, V) = \text{Softmax}(\mathcal{R})V, \end{aligned} \tag{21}$$

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202501_11(1).0002

where $\alpha_1\alpha_2\cdots\alpha_N = HW$, $\beta_1\beta_2\cdots\beta_N = HW$, and $\sum_{i=1}^{N}\alpha_i\beta_i << HW \times HW$. With the reduced size of parameters, it can avoid the over-fitting problem.

Mixture Tensor Synthesizer With the above four kinds of tensor synthesizer models been introduced, a compounded model could be proposed by mixing the synthesizers in additive fashion. For instance:

$$Y = \text{Softmax}(\theta_1 S_1(\mathcal{X}) + \cdots + \theta_M S_M(\mathcal{X}))V, \tag{22}$$

where θ_i are the learnable weights, and $\sum_{i=1}^{M}\theta_i = 1$. S_i are the different synthesizer models. To be specific, the Mixture Tensor Synthesizer in the experiments is composed of Tensor Random Factorized Synthesizer and Tensor Dense Synthesizer, which could be represented as follows:

$$Y = \text{Softmax}(\theta_1\mathcal{R} + \theta_2 Z)V, \tag{23}$$

where $\mathcal{R}$ and Z are defined in Eq. (21) and Eq. (11), respectively. By employing more than one kind of synthesizer, it is possible to achieve better performance.

4. EXPERIMENTS

In this section, the performance of our proposed STT module series on vision tasks is thoroughly investigated on image classification and image captioning. The experiments focus on the performance of the robustness of the model. For convenience, we abbreviate the proposed STT series and comparison methods. The details of information are listed in Table 1. Matrix Dense Synthesizer being the basic synthesizer proposed in [16], and Tensor Dense Synthesizer being our basic tensor synthesizer for vision tasks, Tensor Height Synthesizer being the tensor synthesizer with an attention map full rank in height, and Mixture Tensor Synthesizer being the synthesizers mixed in additive fashion.

Table 1. Abbreviation of model zoo

Abbreviation	Combination
None	Convolution Baseline
CT	CNN + Transformer [44]
CL	CNN + Linformer [40]
SD	CNN + Matrix Dense Synthesizer [16]
STT	CNN + Tensor Dense Synthesizer
SR	CNN + Tensor Random Synthesizer
STTH	CNN + Tensor Height Synthesizer
STTW	CNN + Tensor Width Synthesizer
FSD	CNN + Tensor Dense Factorized Synthesizer
FSR	CNN + Tensor Random Factorized Synthesizer
MS	CNN + Mixture Tensor Synthesizers

4.1. Image Classification

4.1.1 Implementation Details

STT series are validated on the benchmark image dataset CIFAR10 [45] for classification. CIFAR10 is composed of 10 classes, and it has 50000 images for training, 10000 for testing. To

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202501_11(1).0002

validate the functionality of the STT series as attention modules applied to a deep learning framework, we employed three different perturbations to test the model robustness of above all approaches. The applied perturbations include Gaussian noise, image rotations and flipping.

1) Gaussian noise: random noise is added on each pixel of the input image, and we want to test whether the attention computed on the global scale outperforms the split ones.

2) Rotations and flips: once the rotations or flips are exerted, the order among channels would change correspondingly, while STT series treat the feature map as a whole.

To ensure fair comparisons with traditional self-attention modules, we use a two-layer CNN as the baseline and evaluate performance based on classification accuracy. Besides, we select Transformer [44], Linformer [40], and Matrix Dense Synthesizer [16] to test their robustness in the same baseline structure.

The Transformer and Linformer modules employ the dot-product to compute the similarity between token to token. And Matrix Dense Synthesizer also applies the synthesizer mechanism to replace the dot production, but it doesn't leverage the basic tensor synthesizer in their module. Through the comparison, we could prove our STT series satisfy the requirement to become a valuable alternative to the dot-product-based self-attention model. Moreover, it has better robustness performance.

4.1.2 Quantitative Analysis on Robust Classification

According to our proposed STT series, one of the advantages is that they possess strong robustness against external perturbations. The reason is that they calculate attention maps without splitting the feature maps. Thus, the baseline model reduces the reliance on the input, and the underlying structure of the feature map has more possibility to be preserved.

Our STT series has a more negligible effect from external perturbation on the raw input than the dot-production-based self-attention modules. Hence, to evaluate the robustness validation for our proposed STT series, we conduct the image classification task to compare the STT series with traditional self-attention modules.

Gaussian Noise As a common statistical noise caused by sensor during sampling or transmission, Gaussian noise is a widespread type of disturbance for digital images. Varied levels of Gaussian noises are utilized to corrupt the images for model robustness evaluation, with detailed settings and evaluation results provided in Table 2.

Table 2. Accuracy of image classification with relatively small gaussian noise

Noise	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09	0.10
None	45.96	34.35	27.8	22.83	20.01	17.86	16.23	15.07	14.45	13.76
CT	47.02	35.83	29.21	24.7	21.49	19.33	17.41	16.51	15.42	14.8
CL	41.51	28.54	21.69	18.56	16.16	14.53	13.57	12.41	12.17	11.5
SD	48.71	**37.79**	31.54	27.3	23.7	22.23	20.18	18.92	17.45	17.03
SR	45.93	34.02	27.06	23.01	19.91	17.36	16.02	14.93	13.93	13.4
FSR	46.51	37.28	31.69	27.56	24.14	22.6	20.62	19.75	18.25	17.76
FSD	**68.9**	37.5	30.42	25.89	22.85	20.7	19.33	17.91	16.87	16.07
STT	47.29	36.81	**31.98**	**28.11**	**25.49**	**23.75**	**22.39**	**21.44**	**20.09**	**19.54**
STTH	44.66	30.92	23.26	19.58	17.08	15.12	14.04	13.35	12.61	12.26
STTW	42.2	30.21	23.65	19.92	17.53	16.41	15.18	14.65	14.33	13.64
MS	48.2	36.8	29.55	25.13	21.45	19.34	17.74	16.61	15.76	15.26

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202501_11(1).0002

As shown in Table 2, the accuracy of the classification decreases with the Gaussian noise increasing. From the comparison of the two series of methods, it can be concluded that as the variance in the Gaussian distribution increases, the STT series exhibits greater noise resistance than models utilizing dot-production. When the variance of the Gaussian noise is 0.01, the FSD method of the STT series achieves the highest accuracy of 68.9, significantly outperforming other methods. As the noise variance increases, the STT method shows an outstanding advantage, achieving the best performance across all methods when the noise variance ranges from 0.03 to 0.1. Among the STT series, the worse performance of STTH and STTW might result from inferior representation because the information is only gathered from one dimension of the feature map compared to the transformer. Compared with the modules employing dot-product, synthesizing modules appear higher accuracy when Gaussian noise influences them.

Robustness against Flipping Flipping is common in image augmentation process, which could be categorized into horizon flipping, vertical flipping, and flipping both in horizon and vertical. The robustness evaluation results against flipping are recorded in Table 3.

Table 3. Accuracy of images classification with flips

Module	horizon	vertical	horizon + vertical
None	64.74	25.18	25.08
CT	64.93	25.52	25.45
CL	65.31	25.2	25.05
SD	65.7	25.08	25.21
SR	65.86	26.06	25.72
FSR	65.08	24.63	25.14
FSD	**66.28**	26.41	26.58
MS	65.6	25.75	25.99
STT	66.09	27.16	27.32
STTH	66.09	**29.29**	**28.9**
STTW	65.77	27.45	27.46

From Table 3, we observe a significant decrease in accuracy once the vertical transformation is applied. Comparing the synthesizing modules with those employing dot-product, there is no significant difference between the two approaches. It demonstrates that the synthesizer modules are not more vulnerable under the flipping noise. Even though all the models have severe impair from vertical flipping noise, the STT series has slightly better accuracy.

Images with Rotations We apply rotations (Unit: Degrees) to the original images to evaluate the robustness with the model zoo. Detailed settings and results are recorded in Table 4.

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202501_11(1).0002

Table 4. Accuracy of image classification with rotations

Rotation	30	60	90	120	150	180	210	240	270	300	330
None	37.58	26.51	23.07	21.12	19.4	19.39	18.68	18.89	18.27	17.74	17.4
CT	36.54	36.57	23.27	21.25	19.75	19.44	19.7	18.61	18.19	18.46	17.75
CL	37.53	26.86	23.61	21.5	20.16	18.61	19.19	17.95	17.62	17.95	17.95
SD	37.9	**38.64**	23.24	21.79	19.26	19.45	19.36	18.51	17.55	17.6	17.98
SR	33.72	24.7	22.06	19.88	19	18.17	18.03	17.22	17.03	17.23	16.79
FSR	37.04	26.86	23.1	20.69	19.72	18.69	18.7	18.27	17.69	17.17	17.08
FSD	39.27	28.85	23.55	22.85	20.37	19.85	19.95	18.71	18.86	18.34	17.88
MS	37.91	26.93	22.96	21.74	19.61	18.89	18.76	18.01	18.06	17.62	17.65
STT	38.71	38.36	24.51	21.8	20.97	20.36	20.89	19.93	19.86	19.12	20
STTH	42.32	30.44	26.28	**24.53**	22.17	22.18	**22.37**	**21.83**	**20.85**	20.07	**20.16**
STTW	**45.53**	31.72	**27.21**	24	**22.39**	**22.23**	22.03	21.27	20.61	**20.31**	19.89

In Table 4, it is obvious that the classification accuracy of the whole model zoo is impaired when rotation increases. However, the accuracy of the STT series represents a similar decreasing tendency. It proves that the STT series has the same performance as dot-production models when they are suffering the rotation noise.

Despite the STTH and STTW modules, other approaches don't reveal the resistance of rotation noise. While transformer weakens the baseline CNN structure, STTH and STTW have a relatively better accuracy performance. Based on such result, we make an assumption that the full-rank attention map to give them a better robustness against the effect of rotation. This assumption could also explain that the STT has a relatively better result, because it also has a full-rank attention map.

4.2. Image Captioning

4.2.1 Implementation Details

We report our results on COCO-2014 [46] dataset with 82,783 images for training, 40,504 images for validation and 40,775 images for testing. The results are reported under standard evaluation protocol, where captioning metrics include full set of BLEU [47], along with METEOR [48], CIDEr [49], ROUGE [50]. Most of them are based on n-gram to evaluate the quality of the generated sentences with ground truths, and they complement each other to yield fairer results.

4.2.2 Quantitative Analysis of Image Captioning

The performance quantified by the seven metrics is summarized in Table 5, from which we can see that both LSTM-based and transformer-based image captioning framework perform worse than STT series for all seven metrics. It is interesting that LSTM-based model appears to be inferior to transformer-based models, given that transformer frees the model from sequential input dependence. Furthermore, all of modules from STT series tend to improve the results than traditional methods. Besides, MS (Tensor Mixture Synthesizer) seems to achieve most of the best results among the model zoo.

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202501_11(1).0002

Table 5. Image captioning performance with STT series

Meric	BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR	ROUGE	CIDEr
LSTM	0.7191	0.5153	0.3468	0.2339	0.2725	0.5699	0.8128
Transformer	0.7077	0.5035	0.3392	0.2289	0.2692	0.5621	0.7970
SD	0.7921	0.6566	**0.5386**	**0.4457**	0.3686	0.7211	**1.2099**
SR	0.7888	0.6542	0.5359	0.4433	0.3688	**0.7213**	1.2040
MS	**0.7925**	**0.6569**	0.5383	0.4451	**0.3691**	0.7225	0.2104
FSR	0.7876	0.6505	0.5293	0.4345	0.3646	0.7159	1.1793

Moreover, in comparison with baseline caption models with LSTM and Transformer, both SR (Tensor Random Synthesizer) and FSR (Tensor Factorized Random Synthesizer) yield better results with less time for testing, and fewer parameters. The alignments in transformer are conducted on one side of the feature map while dense synthesizer enables a competitive attention for three sides of the feature tensor, which might result in the overall performance superiority of STT against the baselines.

5. CONCLUSION

In this paper, we present a self-attention plug-in module with its variants, named the series of STT. It employs the tensor transformation for self-alignment to replace the traditional dot-product multiplication in the self-attention module. Through this novel approach, the shortage of exhaustive and redundant computation caused by dot-production is dramatically mitigated. In our image caption experiments, the STT series achieve most of the best results with less time in testing, and use fewer parameters. Moreover, because the synthesizing mechanism notably reduces the input feature's reliance, the underlying structure is preserved. The STT series strongly facilitate the robustness of the baseline model. In image classification experiments, the STT series represent the competitive robustness encountering multiple external perturbations. It demonstrates the STT series have the potential to improve the robustness, lower the overfitting, and the flexibility to be extended to more deep learning architectures.

ACKNOWLEDGEMENTS

This paper was supported by General Program of Shanghai Natural Science Foundation (Grant No. 23ZR1419300), Science and Technology Commission of Shanghai Municipality (Grant No. 22DZ2229004)

REFERENCES

[1] Roy, A.; Saffar, M.; Vaswani, A.; Grangier, D. Efficient Content-Based Sparse Attention with Routing Transformers. Trans. Assoc. Comput. Linguistics 2021, 9, 53–68.

[2] Zaheer, M.; Guruganesh, G.; Dubey, A.; Ainslie, J.; Alberti, C.; Ontanon, S.; Pham, P.; Ravula, A.; Wang, Q.; Yang, L.; et al. Big Bird: Transformers for Longer Sequences. In Proceedings of the NeurIPS, 2020.

[3] Kitaev, N.; Kaiser, L.; Levskaya, A. Reformer: The Efficient Transformer. In Proceedings of the ICLR, 2020.

[4] Wu, Q.; Lan, Z.; Gu, J.; Yu, Z. Memformer: The Memory-Augmented Transformer, 2020, [arXiv:cs.CL/2010.06891].

[5] Choromanski, K.M.; Likhosherstov, V.; Dohan, D.; Song, X.; Gane, A.; Sarlós, T.; Hawkins, P.; Davis, J.Q.; Mohiuddin, A.; Kaiser, L.; et al. Rethinking Attention with Performers. In Proceedings of the ICLR, 2021.

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202501_11(1).0002

[6] Rae, J.W.; Potapenko, A.; Jayakumar, S.M.; Hillier, C.; Lillicrap, T.P. Compressive Transformers for Long-Range Sequence Modelling. In Proceedings of the ICLR, 2020.

[7] Zhu, X.; Su, W.; Lu, L.; Li, B.; Wang, X.; Dai, J. Deformable DETR: Deformable Transformers for End-to-End Object Detection. In Proceedings of the ICLR, 2021.

[8] Huang, L.; Wang, W.; Chen, J.; Wei, X. Attention on Attention for Image Captioning. In Proceedings of the ICCV, 2019, pp. 4633–4642.

[9] Fahim, S.R.; Sarker, Y.; Sarker, S.K.; Sheikh, M.R.I.; Das, S.K. Self attention convolutional neural network with time series imaging based feature extraction for transmission line fault detection and classification. Electric Power Systems Research 2020, 187, 106437. https://doi.org/https://doi.org/10.1016/j.epsr.2020.106437.

[10] Tong, W.; Guan, X.; Zhang, M.; Li, P.; Ma, J.; Wu, E.Q.; Zhu, L.M. Edge-assisted epipolar transformer for industrial scene reconstruction. IEEE Transactions on Automation Science and Engineering 2024.

[11] Mnih, V.; Heess, N.; Graves, A.; et al. Recurrent models of visual attention. In Proceedings of the NeurIPS, 2014, pp. 2204–2212.

[12] Ba, J.; Mnih, V.; Kavukcuoglu, K. Multiple Object Recognition with Visual Attention. In Proceedings of the ICLR, 2015.

[13] Wang, F.; Jiang, M.; Qian, C.; Yang, S.; Li, C.; Zhang, H.; Wang, X.; Tang, X. Residual Attention Network for Image Classification. In Proceedings of the CVPR, 2017.

[14] Hu, J.; Shen, L.; Sun, G. Squeeze-and-Excitation Networks. In Proceedings of the CVPR, 2018, pp. 7132–7141.

[15] Woo, S.; Park, J.; Lee, J.Y.; Kweon, I.S. CBAM: Convolutional Block Attention Module. In Proceedings of the ECCV, 2018.

[16] Tay, Y.; Bahri, D.; Metzler, D.; Juan, D.C.; Zhao, Z.; Zheng, C. Synthesizer: Rethinking Self-Attention in Transformer Models, 2020, [arXiv:cs.CL/2005.00743].

[17] Dong, Y.; Cordonnier, J.; Loukas, A. Attention is not all you need: pure attention loses rank doubly exponentially with depth. In Proceedings of the ICML, 2021, Vol. 139, pp. 2793–2803.

[18] Liu, C.; Mao, J.; Sha, F.; Yuille, A.L. Attention Correctness in Neural Image Captioning. In Proceedings of the AAAI, 2017, pp. 4176–4182.

[19] Wang, Y.; Ma, N.; Guo, Z. Machine Reading Comprehension Model Based on Fusion of Mixed Attention. Applied Sciences 2024, 14. https://doi.org/10.3390/app14177794.

[20] Hu, J.; Shen, L.; Albanie, S.; Sun, G.; Vedaldi, A. Gather-Excite: Exploiting Feature Context in Convolutional Neural Networks. In Proceedings of the NeurIPS, 2018, pp. 9423–9433.

[21] Chen, L.; Zhang, H.; Xiao, J.; Nie, L.; Shao, J.; Liu, W.; Chua, T.S. Sca-cnn: Spatial and channel-wise attention in convolutional networks for image captioning. In Proceedings of the CVPR, 2017, pp. 5659–5667.

[22] Fu, J.; Liu, J.; Tian, H.; Li, Y.; Bao, Y.; Fang, Z.; Lu, H. Dual Attention Network for Scene Segmentation. In Proceedings of the CVPR, 2019, pp. 3146–3154.

[23] Bello, I.; Zoph, B.; Vaswani, A.; Shlens, J.; Le, Q.V. Attention augmented convolutional networks. In Proceedings of the ICCV, 2019, pp. 3286–3295.

[24] Parmar, N.; Ramachandran, P.; Vaswani, A.; Bello, I.; Levskaya, A.; Shlens, J. Stand-Alone Self-Attention in Vision Models. In Proceedings of the NeurIPS, 2019, pp. 68–80.

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202501_11(1).0002

[25] Cordonnier, J.B.; Loukas, A.; Jaggi, M. On the Relationship between Self-Attention and Convolutional Layers. In Proceedings of the ICLR, 2020.

[26] Zhao, H.; Jia, J.; Koltun, V. Exploring self-attention for image recognition. In Proceedings of the CVPR, 2020, pp. 10076–10085.

[27] Maurício, J.; Domingues, I.; Bernardino, J. Comparing Vision Transformers and Convolutional Neural Networks for Image Classification: A Literature Review. Applied Sciences 2023.

[28] Hu, H.; Gu, J.; Zhang, Z.; Dai, J.; Wei, Y. Relation Networks for Object Detection. In Proceedings of the CVPR, 2018, pp. 3588–3597.

[29] Carion, N.; Massa, F.; Synnaeve, G.; Usunier, N.; Kirillov, A.; Zagoruyko, S. End-to-End Object Detection with Transformers. In Proceedings of the ECCV, 2020, Vol. 12346, pp. 213–229.

[30] Wang, H.; Zhu, Y.; Green, B.; Adam, H.; Yuille, A.; Chen, L.C. Axial-DeepLab: Stand-Alone Axial-Attention for Panoptic Segmentation, 2020, [arXiv:cs.CV/2003.07853].

[31] Ondeng, O.; Ouma, H.; Akuon, P. A Review of Transformer-Based Approaches for Image Captioning. Applied Sciences 2023, 13. https://doi.org/10.3390/app131911103.

[32] Tong, W.; Zhang, M.; Zhu, G.; Xu, X.; Wu, E.Q. Robust Depth Estimation Based on Parallax Attention for Aerial Scene Perception. IEEE Transactions on Industrial Informatics 2024.

[33] Parmar, N.; Vaswani, A.; Uszkoreit, J.; Kaiser, L.; Shazeer, N.; Ku, A.; Tran, D. Image transformer. In Proceedings of the ICML, 2018, pp. 4055–4064.

[34] Ho, J.; Kalchbrenner, N.; Weissenborn, D.; Salimans, T. Axial attention in multidimensional transformers. arXiv preprint arXiv:1912.12180 2019.

[35] Chen, M.; Radford, A.; Child, R.; Wu, J.; Jun, H.; Luan, D.; Sutskever, I. Generative pretraining from pixels. In Proceedings of the ICML, 2020, pp. 1691–1703.

[36] Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In Proceedings of the ICLR, 2021.

[37] Huang, L.; Yuan, Y.; Guo, J.; Zhang, C.; Chen, X.; Wang, J. Interlaced sparse self-attention for semantic segmentation. arXiv preprint arXiv:1907.12273 2019.

[38] Huang, Z.; Wang, X.; Huang, L.; Huang, C.; Wei, Y.; Liu, W. CCNet: Criss-Cross Attention for Semantic Segmentation. In Proceedings of the ICCV, 2019, pp. 603–612.

[39] Chen, Y.; Kalantidis, Y.; Li, J.; Yan, S.; Feng, J. Aˆ2-Nets: Double Attention Networks. In Proceedings of the NeurIPS, 2018, pp. 350–359.

[40] Wang, S.; Li, B.Z.; Khabsa, M.; Fang, H.; Ma, H. Linformer: Self-Attention with Linear Complexity, 2020, [arXiv:cs.LG/2006.04768].

[41] De Lathauwer, L.; De Moor, B.; Vandewalle, J. A multilinear singular value decomposition. SIAM journal on Matrix Analysis and Applications 2000, 21, 1253–1278.

[42] Van Loan, C.F. The ubiquitous Kronecker product. Journal of computational and applied mathematics 2000, 123, 85–100.

[43] Seibert, M.; Wörmann, J.; Gribonval, R.; Kleinsteuber, M. Learning co-sparse analysis operators with separable structures. IEEE Transactions on Signal Processing 2015, 64, 120–130.

[44] Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In Proceedings of the NeurIPS, 2017, pp. 5998–6008.

[45] Krizhevsky, A.; Hinton, G.; et al. Learning multiple layers of features from tiny images; Master Thesis, 2009.

ISSN: 2472-3703 DOI: 10.6911/WSRJ.202501_11(1).0002

[46] Chen, X.; Fang, H.; Lin, T.Y.; Vedantam, R.; Gupta, S.; Dollár, P.; Zitnick, C.L. Microsoft coco captions: Data collection and evaluation server. arXiv preprint arXiv:1504.00325 2015.

[47] Karpathy, A.; Fei-Fei, L. Deep visual-semantic alignments for generating image descriptions. In Proceedings of the CVPR, 2015, pp. 3128–3137.

[48] Denkowski, M.; Lavie, A. Meteor universal: Language specific translation evaluation for any target language. In Proceedings of the Proceedings of the ninth workshop on statistical machine translation, 2014, pp. 376–380.

[49] Vedantam, R.; Lawrence Zitnick, C.; Parikh, D. Cider: Consensus-based image description evaluation. In Proceedings of the CVPR, 2015, pp. 4566–4575.

[50] Lin, C.Y. Rouge: A package for automatic evaluation of summaries. In Proceedings of the Text summarization branches out, 2004, pp. 74–81.